SERVLETS - COOKIES HANDLING

Cookies are text files stored on the client computer and they are kept for various information tracking purpose. Java Servlets transparently supports HTTP cookies.

There are three steps involved in identifying returning users:

- Server script sends a set of cookies to the browser. For example name, age, or identification number etc.
- Browser stores this information on local machine for future use.
- When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

This chapter will teach you how to set or reset cookies, how to access them and how to delete them.

The Anatomy of a Cookie:

Cookies are usually set in an HTTP header (although JavaScript can also set a cookie directly on a browser). A servlet that sets a cookie might send headers that look something like this:

```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name=xyz; expires=Friday, 04-Feb-07 22:03:38 GMT;
                path=/; domain=tutorialspoint.com
Connection: close
Content-Type: text/html
```

As you can see, the Set-Cookie header contains a name value pair, a GMT date, a path and a domain. The name and value will be URL encoded. The expires field is an instruction to the browser to "forget" the cookie after the given time and date.

If the browser is configured to store cookies, it will then keep this information until the expiry date. If the user points the browser at any page that matches the path and domain of the cookie, it will resend the cookie to the server. The browser's headers might look something like this:

```
GET / HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.6 (X11; I; Linux 2.2.6-15apmac ppc)
Host: zink.demon.co.uk:1126
Accept: image/gif, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: name=xyz
```

A servlet will then have access to the cookie through the request method *request.getCookies()* which returns an array of *Cookie* objects.

Servlet Cookies Methods:

Following is the list of useful methods which you can use while manipulating cookies in servlet.

S.N.	Method & Description

1	**public void setDomain(String pattern)** This method sets the domain to which cookie applies, for example tutorialspoint.com.
2	**public String getDomain()** This method gets the domain to which cookie applies, for example tutorialspoint.com.
3	**public void setMaxAge(int expiry)** This method sets how much time (in seconds) should elapse before the cookie expires. If you don't set this, the cookie will last only for the current session.
4	**public int getMaxAge()** This method returns the maximum age of the cookie, specified in seconds, By default, -1 indicating the cookie will persist until browser shutdown.
5	**public String getName()** This method returns the name of the cookie. The name cannot be changed after creation.
6	**public void setValue(String newValue)** This method sets the value associated with the cookie.
7	**public String getValue()** This method gets the value associated with the cookie.
8	**public void setPath(String uri)** This method sets the path to which this cookie applies. If you don't specify a path, the cookie is returned for all URLs in the same directory as the current page as well as all subdirectories.
9	**public String getPath()** This method gets the path to which this cookie applies.
10	**public void setSecure(boolean flag)** This method sets the boolean value indicating whether the cookie should only be sent over encrypted (i.e. SSL) connections.
11	**public void setComment(String purpose)** This method specifies a comment that describes a cookie's purpose. The comment is useful if the browser presents the cookie to the user.
12	**public String getComment()** This method returns the comment describing the purpose of this cookie, or null if the cookie has no comment.

Setting Cookies with Servlet:

Setting cookies with servlet involves three steps:

(1) Creating a Cookie object: You call the Cookie constructor with a cookie name and a cookie value, both of which are strings.

```
Cookie cookie = new Cookie("key","value");
```

Keep in mind, neither the name nor the value should contain white space or any of the following characters:

```
[ ] ( ) = , " / ? @ : ;
```

(2) Setting the maximum age: You use setMaxAge to specify how long (in seconds) the cookie should be valid. Following would set up a cookie for 24 hours.

```
cookie.setMaxAge(60*60*24);
```

(3) Sending the Cookie into the HTTP response headers: You use **response.addCookie** to add cookies in the HTTP response header as follows:

```
response.addCookie(cookie);
```

Example:

Let us modify our Form Example to set the cookies for first and last name.

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class HelloForm extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      // Create cookies for first and last names.
      Cookie firstName = new Cookie("first_name",
                     request.getParameter("first_name"));
      Cookie lastName = new Cookie("last_name",
                     request.getParameter("last_name"));

      // Set expiry date after 24 Hrs for both the cookies.
      firstName.setMaxAge(60*60*24);
      lastName.setMaxAge(60*60*24);

      // Add both the cookies in the response header.
      response.addCookie( firstName );
      response.addCookie( lastName );

      // Set response content type
      response.setContentType("text/html");

      PrintWriter out = response.getWriter();
      String title = "Setting Cookies Example";
      String docType =
      "<!doctype html public \"-//w3c//dtd html 4.0 " +
      "transitional//en\">\n";
      out.println(docType +
                "<html>\n" +
                "<head><title>" + title + "</title></head>\n" +
                "<body bgcolor=\"#f0f0f0\">\n" +
                "<h1 align=\"center\">" + title + "</h1>\n" +
                "<ul>\n" +
                "  <li><b>First Name</b>: "
                + request.getParameter("first_name") + "\n" +
                "  <li><b>Last Name</b>: "
                + request.getParameter("last_name") + "\n" +
                "</ul>\n" +
                "</body></html>");
  }
}
```

Compile above servlet **HelloForm** and create appropriate entry in web.xml file and finally try following HTML page to call servlet.

```
<html>
<body>
<form action="HelloForm" method="GET">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Keep above HTML content in a file Hello.htm and put it in <Tomcat-installation-directory>/webapps/ROOT directory. When you would access *http://localhost:8080/Hello.htm*, here is the actual output of the above form.

First Name:

Last Name:

Try to enter First Name and Last Name and then click submit button. This would display first name and last name on your screen and same time it would set two cookies firstName and lastName which would be passed back to the server when next time you would press Submit button.

Next section would explain you how you would access these cookies back in your web application.

Reading Cookies with Servlet:

To read cookies, you need to create an array of *javax.servlet.http.Cookie* objects by calling the **getCookies( )** method of *HttpServletRequest*. Then cycle through the array, and use getName() and getValue() methods to access each cookie and associated value.

Example:

Let us read cookies which we have set in previous example:

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class ReadCookies extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      Cookie cookie = null;
   Cookie[] cookies = null;
      // Get an array of Cookies associated with this domain
      cookies = request.getCookies();

   // Set response content type
      response.setContentType("text/html");

      PrintWriter out = response.getWriter();
      String title = "Reading Cookies Example";
      String docType =
      "<!doctype html public \"-//w3c//dtd html 4.0 " +
      "transitional//en\">\n";
      out.println(docType +
                "<html>\n" +
                "<head><title>" + title + "</title></head>\n" +
                "<body bgcolor=\"#f0f0f0\">\n" );
```

```
      if( cookies != null ){
         out.println("<h2> Found Cookies Name and Value</h2>");
         for (int i = 0; i < cookies.length; i++){
            cookie = cookies[i];
            out.print("Name : " + cookie.getName( ) + ",  ");
            out.print("Value: " + cookie.getValue( )+" <br/>");
         }
      }else{
          out.println(
            "<h2>No cookies founds</h2>");
      }
      out.println("</body>");
      out.println("</html>");
   }
}
```

Compile above servlet **ReadCookies** and create appropriate entry in web.xml file. If you would have set first_name cookie as "John" and last_name cookie as "Player" then running *http://localhost:8080/ReadCookies* would display the following result:

Found Cookies Name and Value

Name : first_name, Value: John
Name : last_name, Value: Player

Delete Cookies with Servlet:

To delete cookies is very simple. If you want to delete a cookie then you simply need to follow up following three steps:

- Read an already exsiting cookie and store it in Cookie object.
- Set cookie age as zero using **setMaxAge()** method to delete an existing cookie.
- Add this cookie back into response header.

Example:

Following example would delete and existing cookie named "first_name" and when you would run ReadCookies servlet next time it would return null value for first_name.

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class DeleteCookies extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      Cookie cookie = null;
   Cookie[] cookies = null;
      // Get an array of Cookies associated with this domain
      cookies = request.getCookies();

   // Set response content type
      response.setContentType("text/html");
```

```
      PrintWriter out = response.getWriter();
      String title = "Delete Cookies Example";
      String docType =
      "<!doctype html public \"-//w3c//dtd html 4.0 " +
      "transitional//en\">\n";
      out.println(docType +
                "<html>\n" +
                "<head><title>" + title + "</title></head>\n" +
                "<body bgcolor=\"#f0f0f0\">\n" );
       if( cookies != null ){
         out.println("<h2> Cookies Name and Value</h2>");
         for (int i = 0; i < cookies.length; i++){
            cookie = cookies[i];
            if((cookie.getName( )).compareTo("first_name") == 0 ){
                 cookie.setMaxAge(0);
                 response.addCookie(cookie);
                 out.print("Deleted cookie : " +
                             cookie.getName( ) + "<br/>");
            }
            out.print("Name : " + cookie.getName( ) + ",  ");
            out.print("Value: " + cookie.getValue( )+" <br/>");
         }
      }else{
          out.println(
            "<h2>No cookies founds</h2>");
      }
      out.println("</body>");
      out.println("</html>");
   }
}
```

Compile above servlet **DeleteCookies** and create appropriate entry in web.xml file. Now running *http://localhost:8080/DeleteCookies* would display the following result:

Cookies Name and Value

Deleted cookie : first_name
Name : first_name, Value: John
Name : last_name, Value: Player

Now try to run *http://localhost:8080/ReadCookies* and it would display only one cookie as follows:

Found Cookies Name and Value

Name : last_name, Value: Player

You can delete your cookies in Internet Explorer manually. Start at the Tools menu and select Internet Options. To delete all cookies, press Delete Cookies.